Programming in C++

Classes

Curtis Larsen

Utah Tech University—Computing

Spring 2025

Objectives

Objectives:

- Create and call constructors
- Use data member initialization
- Create access control
- Access members (data and function)

Constructors

Declare

Constructor properties:

- Special methods
- Named same as class
- Have no return type (not even void)
- Can be overloaded

```
class Point2D {
public:
  Point2D();
  Point2D(const double x, const double y);
...
protected:
  double mX;
  double mY;
};
```

Implement

Implementation properties:

- Initialize data members, in declaration order
- Defaults, parameters, computation to initialize

```
Point2D::Point2D()
  : mX(0.0), mY(0.0) {
}
Point2D::Point2D(const double x, const double y)
  : mX(0.0), mY(0.0) {
  setX(x);
  setY(y);
}
```

Invoke (Call)

Invoke properties:

- No `()` uses default constructor
- Argument list is matched to constructor parameter list

```
Point2D p; // Point2D object, default constructed.
Point2D q(3.14, 2.71); // Point2D object, value constructed
```

Specifying Access

Specifiers

- `private` - Only the class methods can access (default)
- `protected` - Only class or derived class methods can access
- `public` - Any code can access

```
class Point2D {
public:
  Point2D();
  Point2D(const double x, const double y);
  double getX() const;
  double getY() const;
  void setX(const double x);
  void setY(const double y);
  // straight line distance from the origin
  double distance() const;

protected:
  double mX;
  double mY;
};
```

Member Access

Data Members

- `mX` - raw name, scoping rules apply (`this` is implicit)
- `this->mX` - `this` is explicit member access
- `this` is implicitly declared inside methods

```
double Point2D::getX() const {
  return mX;
}
void Point2D::setX(const double x) {
  if(x >= 0.0) {
    this->mX = x;
  }
}
```

Member Functions (Methods)

- `setX` - raw name, scoping rules apply (`this` is implicit)
- `this->setX` - `this` is explicit member access
- `(*this)->setX` - same meaning as `this->setX`
- `this` is implicitly declared inside methods

```
Point2D::Point2D(const double x, const double y)
  : mX(0.0), mY(0.0) {
  setX(x);
  this->setX(x);
  (*this).setY(y);
}
```